REMARKS

This Amendment and the following remarks are intended to fully respond to the Final Office Action dated January 12, 2005. In that Final Office Action, claims 1-32 were examined, and all claims were rejected. More particularly, claims 1-32 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Gosling (USPN 6,247,044) in view of Logston (USPN 6,687,735). Reconsideration of these rejections, as they might apply to the original and amended claims in view of these remarks, is respectfully requested.

In this Response, claims 1, 11 and 22 are amended herein and claims 9, 20 and 31 are cancelled. Therefore, claims 1-8, 10-19, 21-30 and 32 remain present for examination.

Claim Amendments

Applicants amend claims 1, 11 and 22 herein to include the limitation (from claims 9, 20 and 31 respectively) that each event pipeline is a separate instance that corresponds to a single context object. The claims are further amended to point out that such an instance only exists for the lifetime of its corresponding context object. The specification discusses in several locations, such as in paragraph [0020] (on page 9, lines 1-4) that HTTP requests and their corresponding context objects have a lifetime. Furthermore, paragraph 22 clearly states that an event pipeline generates callbacks for a context object only during the lifetime of the context object. The lifetime of an HTTP request is described further in paragraph [0022] (page 10, lines 4-18) of the specification and it ends with the response being sent by the server to the client. While it is not specifically stated, it is well known to one skilled in the art that an instance, which is taking up memory resources, reaches the end of its lifetime, it is deleted from memory. If it were otherwise, all resources would soon be used up by instances that are no longer in use.

Therefore, the Applicants consider it an inherent aspect of the embodiment claimed in original claims 9, 20 and 31 that each runtime instance exists only during the lifetime of its corresponding as the application clearly states in original claims 9, 20 and 31 that each instance is created to correspond with one context object and, as stated in paragraph [0020], that each context object and HTTP response has a lifetime.

ARGUMENTS IN RESPONSE TO EXAMINER'S RESPONSE

In the Final Office Action, the Examiner maintained his previous rejections that were made in the June 3, 2004 Office Action. The Examiner also provided a very brief response to the Applicants' arguments. The Examiner's response clearly showed that the Examiner does not have a sufficient understanding of the claim elements in question, and that the Examiner has glossed over the limitations in the claim elements without thoroughly addressing them.

For example, the Examiner states that "a servlet object is served as a context object that logically represents an HTTP request" {Final Office Action Paragraph 4.) when, in fact, Gosling states the following about servlet:

> "Each servlet 56 is a piece of software code which is used to dynamically generate information. Each servlet 56 is an instantiated software object waiting to be invoked. Once it is invoked, it dynamically generates information." Gosling, col. 3, lines 17-19.

The Examiner plainly ignores the claim limitation that the context object must represent the HTTP request in question, as the servlets in Gosling are clearly independent of any HTTP request.

As another example, the Examiner states the following:

> "Regarding the applicant's argument about forming an event pipeline, Gosling teaches that after the communication is established between the client request and the servlet object, subsequent request from the client and subsequent responses from the servlet create an event pipeline." Office Action, Section 4.

Applicants submit that communications between a servlet object and a client does not form an event pipeline as claimed, and certainly not an event pipeline with all the limitations as claimed. Here the Examiner plainly ignored the additional limitations on the event pipeline such as that the event pipeline actually processes the context object and that it has "a plurality of request events". See paragraph [0012], ("The context object is processed by the event pipeline, which includes a plurality of request events."). Amended claim 1, now with the limitation originally in claim 9, requires that a separate instance of the event pipeline correspond to the context object while the specification states in paragraph [0025] that, "[e]ach HTTP Module that is registered with one or more request events is instantiated when the pipeline is instantiated."

These limitation were wholly neglected by the Examiner, otherwise the Examiner would not have stated "subsequent requests from the client and subsequent responses from the servlet create an event pipeline" as he did in Paragraph 4 of the Final Office Action. Furthermore, as pointed out in the above discussion regarding the claim amendments, a context object has a specific lifetime and, likewise, so to its corresponding event pipeline.

Applicants' believe that the Examiner has not been thorough in addressing all of the limitations of the claims, and that the references provided by the Examiner do not, upon deeper investigation, show all of the elements as claimed. Applicants' further submit that the Examiner did the same when reviewing our previous arguments and that these arguments remain valid and prove that the original claims are allowable over the cited art. For completeness, the Applicants reproduce their previous arguments below and request that the Examiner consider them again.

RESTATEMENT OF PRIOR ARGUMENTS

For at least the reasons provided below, Applicants believe that the claims are patentable distinguishable over the Gosling and Logston references.

Claim Rejections - 35 U.S.C. § 103

Claims 1-32 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Gosling (USPN 6,247,044) in view of Logston (USPN 6,687,735). In the Office Action, Examiner specifically addresses the elements of independent claim 11, noting that similar elements exist in the other independent claims. Applicants are likewise responding by addressing the specific claim elements in claim 11 with the understanding that these arguments equally apply to the other independent claims.

In order to establish *prima facie* obviousness under 35 U.S.C. 103(a), three basic criteria must be met, namely: (1) there must be some suggestion or motivation to combine the references or modify the reference teaching; (2) there must be a reasonable expectation of success; and (3) the reference or references when combined must teach or suggest each claim limitation (Manual of Patent Examining Procedure 2142). Applicants submit that the Office

Action failed to state a *prima facie* case of obviousness, and therefore the burden has not properly shifted to Applicants to present evidence of nonobviousness. Applicants respectfully assert that the Examiner has failed to establish a *prima facie* case of obviousness because the reference fails to disclose or suggest all of the limitations of the pending claims as discussed below.

Neither Gosling nor Logston Anticipate "forming an event pipeline"

The Examiner cites Gosling, Col. 5, lines 42-43 as anticipating the "forming an event pipeline corresponding to the context object" element of the claimed invention. Office Action, Page 3, lines 3-8. Applicants respectfully disagree with the Examiner's analysis of Gosling.

Gosling is directed to a system in which a plurality of servlet objects is used to replace the use of a monolithic server application. Gosling specifically states that the servlets are continuously operating and waiting for calls from the server computer. See, Gosling, Col. 2, lines 8-11 ("The servlet objects operate in a continual loop until invoked. Thus, there is no startup overhead associated with execution of the servlet objects."). See also, Gosling, Col. 3, lines 17-30 ("Each servlet 56 is an instantiated software object waiting to be invoked... However, unlike a CGI script, a servlet object of the present invention is instantiated at server start-up. Thus, the servlet can be thought of as operating in a continual loop waiting to be executed. Observe that after instantiation there is no computational start-up expense when the servlet is called."). The Examiner's citation only applies to the situation in which a necessary servlet is not locally available. See, Gosling Col. 5, lines 40-55. In addition, Gosling goes on to state that, once instantiated, the servlets loop continuously until the server is shut off or a destroy method is called. See, Col. 6, lines 45-46 ("Once instantiated, a servlet loops until the server is shut off or a destroy method is called on the servlet by the server.").

As the above discussion shows, in Gosling none of the servlets correspond to a context object. The servlets (even those retrieved from a remote server as necessary), once instantiated, wait to be called regardless of the requests that are received. While servlets may correspond to a request type, they do not correspond to any specific request. Thus, a servlet cannot be

considered to correspond directly to a specific request, and therefore do not anticipate an event pipeline corresponding to a context object.

It should be noted that Logston similarly does not disclose an "event pipeline corresponding to a context object." Logston discloses a distributed application in which a client portion (Logston's DACP) and a server portion (Logston's DASP) work in concert as a single application, the benefit of which is to minimize the amount of code transmitted to the client, the work done at the client, and allow easy load-balancing at the server by allowing multiple identical DASPs to be allocated throughout a server farm as necessary. Similar to Gosling, Logston's DASPs exist independently of the DACPs and, indeed, specific DACP requests with the exception that a DASP will be shutdown if it is not servicing (i.e. registered for) at least one DACP.

For the reasons given above, Applicants believe that neither Gosling nor Logston, singly or in combination anticipate the event pipeline element of the independent claims in the pending application. Therefore, Applicants respectfully request that the Examiner withdraw his rejection and find claims 1-32 in a condition for allowance.

Neither Gosling nor Logston Anticipate an "event pipeline having a plurality of request events" or "request event having a corresponding to event"

The Examiner cites Gosling, Col. 7, line 15 as anticipating the "event pipeline having a plurality of request events" element of the claimed invention. Office Action, Page 3, lines 3-8. Applicants respectfully disagree with the Examiner's analysis of Gosling and the invention as claimed.

It appears from the citation provided that the Examiner is equating an HTTP request in Gosling with a request event in an event pipeline in the claimed invention. The present application states in paragraph [0025] that, "The event pipeline provides an event-based architecture that includes a plurality of request events for generating a callback when the request event is raised." Thus, the request events are part of the event pipeline formed corresponding to the context object and not necessarily part of the actual request received. Applicants further

point out that the event pipeline processes the context object. See paragraph [0012], ("The context object is processed by the event pipeline, which includes a plurality of request events.").

The Applicants believe that the Examiner is trying to interpret Gosling's servlet as an event pipeline (which is incorrect for the reasons given above) and then cannot provide support for the distinguishing elements of the event pipeline. This is probably because Gosling states that each HTTP request is mapped to a single servlet, while one servlet may handle multiple requests. See, Gosling, Col. 4, lines 46-49 ("The thread then maps the request to a servlet name (step 78). The servlet may be specified by a URL, in which case the mapping process is direct."). While this servlet may initiate calls to other servlets, Gosling is clear that each HTTP request is mapped to a single servlet. The Applicants' system, on the other hand, forms the event pipeline that then uses as many applications and modules as necessary to process the request events in the event pipeline.

For at least these reasons, Applicants believe that Gosling does not anticipate an "event pipeline having a plurality of request events" in which "request event having a corresponding event" as claimed. Therefore, Applicants respectfully request that the Examiner withdraw the rejection of the independent claims 1, 11 and 22 and all the claims that depend therefrom.

Gosling Teaches Away from "forming an event pipeline"

This may already be clear from the above discussion, but Gosling clearly teaches away from forming an event pipeline as claimed. In numerous locations, Gosling states that the benefit of his system is that the servlets are already running upon receipt of client requests. See, Gosling, Col. 2, lines 10-11 ("Thus, there is no startup overhead associated with execution of the servlet objects."). See also, Gosling, Col. 3, lines 27-30 ("a servlet object of the present invention is instantiated at server start-up. Thus, the servlet can be thought of as operating in a continual loop waiting to be executed. Observe that after instantiation there is no computational start-up expense when the servlet is called." emphasis added).

Gosling would consider forming an event pipeline for each context object formed to be an unreasonable computing overhead because of the additional computation resources necessary for each request. Gosling would categorically object to the Applicants' system as disclosed in

the specification in paragraph [0025] in which "Each HTTP Module that is registered with one or more request events is instantiated when the pipeline is instantiated." Gosling's premise is to have everything instantiated prior to the receipt of the HTTP request. In at least this aspect Gosling's system is diametrically opposed the disclosed system of the Applicants.

Logston Fails to Anticipate "generating" and "initiating" as Claimed

The Examiner asserts that, while Gosling does not disclose or suggest the "generating" or "initiating" elements as claimed, these elements may be found in Logston. The Examiner presents Logston as anticipating the "generating" and "initiating" elements of the claimed invention. The Applicants respectfully disagree.

Logston discloses a distributed application in which a client portion (Logston's DACP) and a server portion (Logston's DASP) form a single application, the benefit of which is to minimize the amount of code transmitted to the client, the work done at the client, and allow easy load-balancing at the server by allowing multiple identical DASPs to be allocated throughout a server farm as necessary. The important structural elements of Logston include that:

- The DACP and DASP are elements of a single application (Col. 7, lines 57-67);
- The DACP is provided by the server to the client as part of the startup process (Col. 8, lines 17-19);
- Every DASP is identical (Col. 8, lines 1-16);
- Each DACP must be assigned (registered) to a DASP for all communications from the DACP (Col. 33, lines 21-27);
- Each DASP is also "registered" with the OS to receive requests from the OS (Col. 33, 10-13) as is well known for any application that must interact with the OS;
- A DASP can service multiple DACPs (Col. 8, lines 1-16);
- A DASP remains in existence from the initial assignment to a DACP until either there is a timeout period during which there are no communications received from a DACP or the last DACP terminates (Col. 34, lines 3-51). Thus, Logston's DASPs may operate forever, under a scenario in which new DACPs are assigned to the DASP as old DACPs terminate.

Applicants believe that the citations provided by the Examiner relate only to the proposition that communication from a client (i.e., the DACP) can be directed to the appropriate DASP running on the server and that the DASP can receive requests from the operating system or other global control programs. This is well known in the art and is a fundamental part of the

client-server model. That Logston uses the same terminology in "registered" and "callback" does not mean that Logston has anticipated the claimed elements.

Applicants believe that Logston does not disclose the "generating" and "initiating" elements of the claimed invention for several reasons. First, the "generating" claim element includes the requirement that the "at least one of an application and a module is registered in association with the request event." Logston's DASP (while capable of receiving communications from any DACP) is registered to receive all communications from a specified DACP, not a request event in an event pipeline.

Furthermore, Logston does not anticipate the "initiating" element in the claimed invention as that element includes the requirement that each application/module initiated be "registered in association with the request event..." Even at Logston's initial startup, Logston does not initiate a module registered in association with a request event of an event pipeline. Rather, Logston initiates a DASP registered in association with a specified client DACP.

The Examiner may be arguing that the DASP's registration with the operating system anticipates the claimed language, but this is also in error because the argument does not take into account that the request events of the claimed invention are part of the event pipeline and are not any event for which an application may be registered. The events disclosed in Logston relate only to the operating system or the OpenCable application, neither of which are an event pipeline formed for a context object.

In sum, while Logston does disclose registering DASPs for DACPs, Logston does not disclose registering anything in association with a request event, much less a request event associated with an event pipeline formed corresponding to a context object. For at least these reasons, Logston does not teach or disclose the claimed elements of "generating" or "initializing" with all their limitations. Therefore, the Applicants respectfully request that the rejection be withdrawn and all pending claims be allowed.

Conclusion

It is believed that no further fees are due with this Response. However, the Commissioner is hereby authorized to charge any deficiencies or credit any overpayment with respect to this patent application to deposit account number 13-2725.

In light of the above remarks, it is believed that the application is now in condition for allowance, and such action is respectfully requested. Should any additional issues need to be resolved, the Examiner is requested to telephone the undersigned to attempt to resolve those issues.

Respectfully submitted,

MERCHANT & GOULD P.C.
P.O. Box 2903
Minneapolis, Minnesota 55402-0903
(303) 357-1639

Date: March 14, 2005

George C. Lewis
Reg. No. 53,214

K:\Clients\40\40062\0250-US-01\Amend & Resp to 1.12.05 FOA.doc